Testes de API: dos manuais aos automatizados

Carolina Santana Louzada

Analista QA - Venturus

Objetivo do curso

Neste curso revisaremos conceitos importantes relacionados à APIs, métodos de teste e como fazer testes manuais e automatizados utilizando Postman e Rest Assured.

Pré-requisitos

- Fundamentos de qualidade de software
- Fundamento de programação -> JAVA
- Fundamentos de automação

Percurso

Aula 1 A API no mundo da qualidade de software

Aula 2 Testes manuais e automatizados com Postman

Aula 3 Testes automatizados com Rest Assured

Dúvidas durante o curso?

> Fórum do curso
> LinkedIn: Carolina Santana Louzada
> Comunidade online (Discord)

SCAN ME

Aula 1

A API no mundo da qualidade de software

// Testes de API: dos manuais aos automatizados

Objetivos

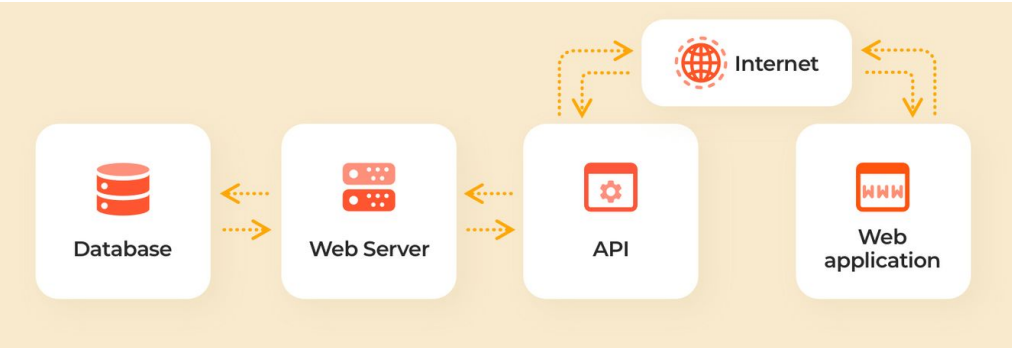
1. Revisar conceitos relacionados à APIs
2. Tipos e tecnologias envolvendo APIs
3. A Importância dos testes de API
4. Planejamento de testes de API e ferramentas
5. Conhecendo a API a ser usada no curso

Aula 1. Etapa 1

Revisando conceitos relacionados à APIs

O que é API?

- Application Programming Interface
- Função: Permitir a comunicação entre aplicações/sistemas a partir de um contrato de serviço.



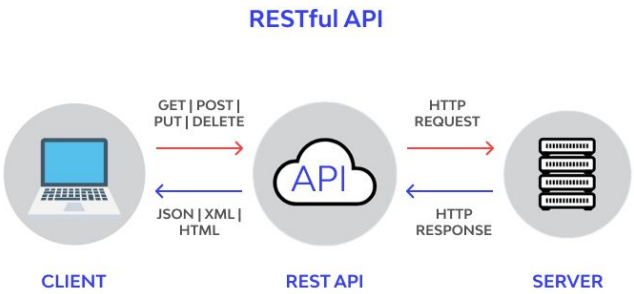
O que é API REST? (redhat.com)

Saiba aqui qual a Diferença entre API e Web Service - Blog Accurate

O que é API?

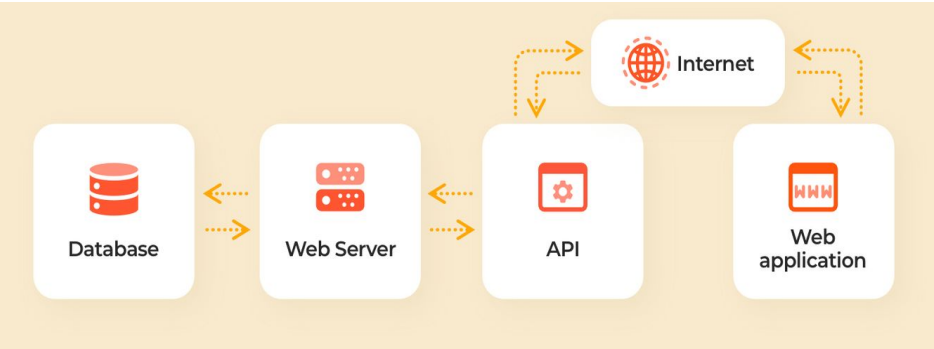
- O que define esse contrato?

A forma como as partes devem se comunicar a partir de uma solicitação e a resposta da mesma.



Fonte: Hakin9

O que é API?



- APIs devem possuir uma boa documentação para que possam ser usadas corretamente

Ex: ViaCEP - Webservice CEP e IBGE gratuito

Benefícios

1. Integração
2. Inovação
3. Expansão
4. Manutenção

Políticas de uso para APIs

- públicas ou abertas 
- privadas ou internas 
- de parceiros 
- compostas 

Tipos/tecnologias

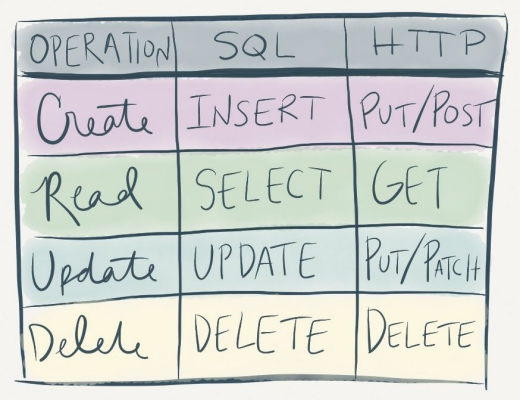
- REST
- RPC
- SOAP
- Baseado em Websocket
- GraphQL

Aula 1. Etapa 2

Tipos e tecnologias envolvendo APIs

REST

- REpresentational State Transfer
- Estilo arquitetural com restrição para criação de aplicações sob o protocolo HTTP
  - Interface Uniforme
  - Cliente-Servidor
  - *Stateless*
  - Uso de camadas
  - Código sob demanda
  - Uso de cache



REST API Definition: What are REST APIs (RESTful APIs)? (astera.com)

REST

- Interface Uniforme
  - Necessário para desacoplar um cliente do servidor
  - Maior visibilidade
  - Controles de Interface:
    - Identificação de recursos
    - Gerenciamento de recursos por representações
    - Comunicações autodescritivas
    - Hipermídia como mecanismo de estado do aplicativo

REST API Definition: What are REST APIs (RESTful APIs)? (astera.com)

Entendendo Hypermedia REST APIs (HATEOAS) – Developer Initiative (wordpress.com)

REST

- Servidor-Cliente
  - Trabalham de forma isolada, com desenvolvimento independente
- *Stateless* / Sem estado
  - As chamadas podem ser feitas de forma independente
  - Cada chamada deve enviar os dados necessários para completar a requisição
- Armazenável em Cache
  - Os dados em uma resposta podem ser armazenados em cache

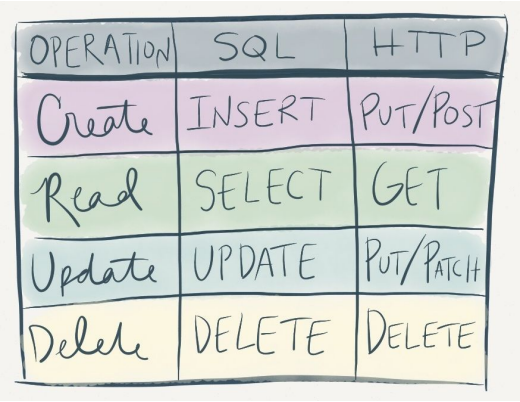
REST

- Sistema em camadas
  - Maior escalabilidade
  - Maior segurança
  - Maior desempenho de cada módulo
- Código sob demanda - opcional
  - Possibilidade dos serviços responderem como representação de um recurso, uma informação executável pelo cliente
  - Diminui o número de recursos a serem implementados

REST

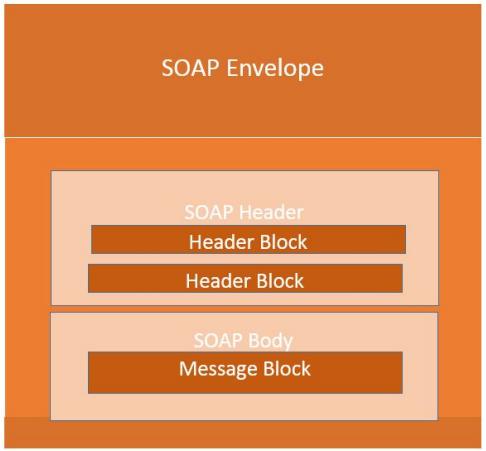
- Pode retornar mensagens em diversos formatos:
  - HTML
  - XML
  - texto
  - JSON

REST API Definition: What are REST APIs (RESTful APIs)? (



SOAP

- Simple Object Access Protocol
- Protocolo de acesso a objetos simples mantido pela W3C e baseado em XML
- Criado com intuito de possibilitar comunicação entre diferentes aplicações e plataformas.
- Suporta protocolos de comunicação como HTTP e SMTP



Protocolo SOAP. SOAP é um protocolo para troca de… | by Gabriel Polo | Medium

SOAP

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 299
SOAPAction: "http://www.w3.org/2003/05/soap-envelope"

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:m="http://www.example.org">
  <soap:Header>
  </soap:Header>
  <soap:Body>
    <m:GetStockPrice>
      <m:StockName>T</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```

WebSocket API

- ➔ Guarda o estado e dados
- ➔ A comunicação é bidirecional, portanto qualquer lado pode enviar mensagem para outra parte
- ➔ Conexão TCP única
- ➔ Bem utilizadas para aplicações IoT/ tempo real

WebSocket vs REST | Learn The 8 Important Differences (educba.com)

Base de comparação	WebSocket	REST
HTTP	Uso na conexão inicial	Comum em RESTful APIs
Comunicação	Bidirecional	Unidirecional
Natureza	Baseado em socket	Baseado em recursos
Cenário	Aplicações de tempo real	Muitas requisições GET
Custo	Baixo	Maior que WebSocket
Performance	Altas cargas	Comunicação ocasional
Estado	Com estado	Sem estado
Dependência	IP e porta somente	HTTP methods

GraphQL

- ➔ Query language para APIs, vindo como alternativa ao REST
- ➔ Não é dependente de algum banco específico
- ➔ São organizadas em termos de tipos e campos, ao invés de endpoints.
- ➔ Facilidade na manutenção

GraphQL | A query language for your API

Introduction to GraphQL | GraphQL

RPC

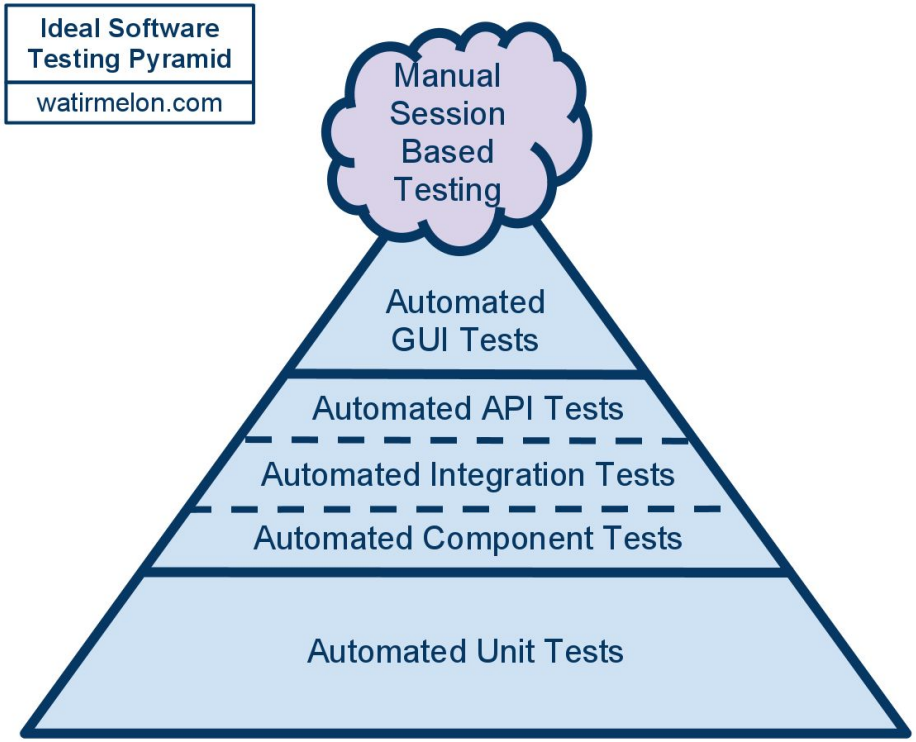
- Remote Procedure Calls
- API é construída com base na definição de métodos que serão chamados via argumentos
- Implementações
  - XML-RPC
  - JSON-RPC
  - SOAP
  - gRPC

Understanding RPC, REST and GraphQL | APIs You Won't Hate (apisyouwonthate.com)

Aula 1. Etapa 3

Importância dos testes de API

Vamos relembrar a pirâmide de testes?



No que consiste um teste de API?

- Processo de monitorar e validar requisições e respostas de uma API, garantindo que se comporte como é esperado.
- A partir da API conseguimos avaliar
  - questões relacionadas a regras de negócio antes de chegarmos ao nível de interface
  - performance
  - integração com outras APIs

No que consiste um teste de API?

- Processo de monitorar e validar requisições e respostas de uma API, garantindo que se comporte como é esperado.
- A partir da API conseguimos avaliar
  - questões relacionadas a regras de negócio antes de chegarmos ao nível de interface.
  - performance
  - integração com outras APIs

Abordagens para testes de API

- ★ Teste funcional: verificamos se a API retorna o que é esperado dada uma requisição
  - status code
  - schema
- ★ Teste de carga: verificamos desempenho a partir de um largo volume de requisições em um curto período
- ★ Detecção de erros: verificamos se API foi bem desenhada e seus erros estão sendo bem monitorados e mensagens estão claras

Abordagens para testes de API

- ★ Segurança: verificamos como a API se comporta e resiste durante ataques.
- ★ Penetração: Verificamos se com pouco conhecimento sobre a API conseguimos atacar uma API.
- ★ Fuzz: Envio de requisições aleatórias para analisar comportamento da API.

Importância de testes de API

- ➔ Garantimos que a API se comporte como deve mediante situações inesperadas.
- ➔ Garantimos que erros não cheguem ao nível de testes de com interface.
- ➔ Avaliamos mais cedo questões relacionadas a desempenho, performance e segurança.
- ➔ Verificamos integração entre APIs.
- ➔ É mais rápido que testes de front-end.

E agora? Se convenceu da importância dos testes de API?

Aula 1. Etapa 4

Planejando testes de API

O que devo pensar antes de iniciar os testes?

- ➔ Qual modelo de ciclo de vida o projeto segue?
- ➔ Quantas pessoas estão na equipe de QA?
- ➔ Qual a priorização de requisitos ?
- ➔ Qual o prazo?
- ➔ Quais processos devemos seguir?
- ➔ Quais ferramentas eu e minha equipe utilizaremos?

1 - Revise especificação da API

- Revisar a especificação da API é importante para:
  - Entender propósito
  - Entender relação com regras de negócio
  - Entender o que é esperado da requisição e na resposta
- A forma como será criada e mantida a especificação deve ser conversada pela equipe

1 - Revise a especificação da API

- Padrão para descrições: OpenAPI - OAI/OpenAPI-Specification: The OpenAPI Specification Repository (github.com)
  - definição de padrões para descrição de HTTP-APIs independente de linguagem
- Swagger: Ferramenta que utiliza o OpenAPI, e constrói uma gama de utilidades e facilidades para geração, manutenção de documentação e testes de API

  Ex: *http://petstore.swagger.io/*

2 - Determine os requerimentos e complexidades

- O que deve ser testado?
  - Status code
  - cabeçalho
  - corpo - > schema

- Qual a complexidade envolvida na codificação dos testes?
  - Infraestrutura
  - Tipos de testes

2 - Determine os requerimentos e complexidades

- Quais abordagens de testes?
- Quais integrações existem com a API?
- Qual conjunto de testes serão automatizados? E quais serão somente manuais?
- Quais cenários prioritários?
  - casos de sucesso (20x)
  - casos de erro/alternativos

3 - Determine a ferramenta de teste manual/automatizada

➔ Ao responder as etapas anteriores a escolha da ferramenta será muito mais fácil!

- Postman/Insomnia
- SoapUI
- Katalon
- Assertible
- Tricentis Tosca
- API Fortress
- Mocklets
- Rest Assured
- Swagger
- Apache JMeter
- APIGee
- Cypress

Que tal pesquisar essas ferramentas?

Aula 1. Etapa 5

Conhecendo a API a ser usada no curso

PetStore

- PetStore - Swagger
- Ferramentas a serem usadas para testes:
  - Postman
  - RestAssured

Aula 2

Testes manuais e automatizados de API com Postman

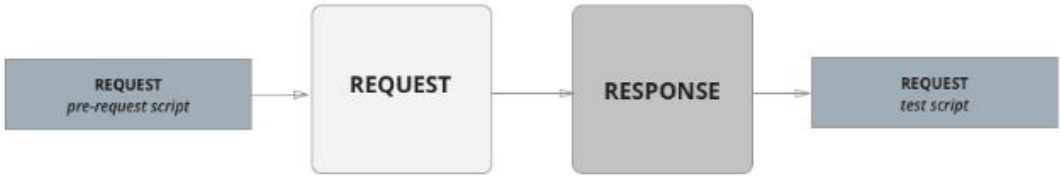
// Testes de API: dos manuais aos automatizados

Objetivos

1. Conhecer e configurar o Postman Download Postman | Get Started for Free
2. Criando a collection
3. Automatizar a collection

Sobre o Postman

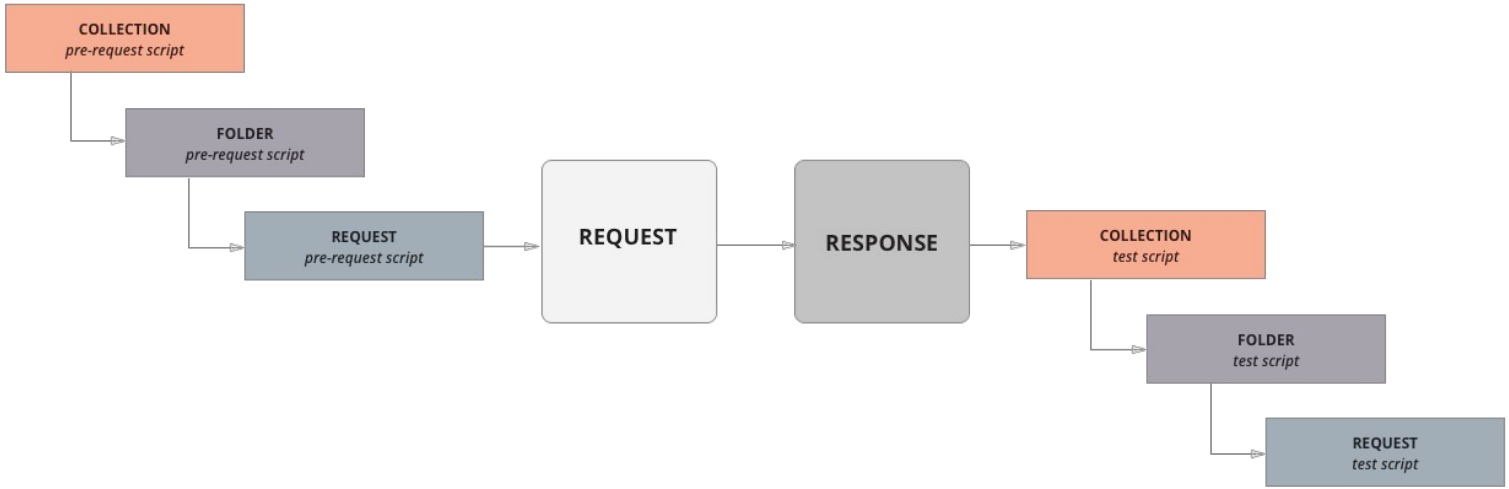
- ★ Poderoso *runtine* baseado em Node.js que permite adicionar comportamento dinâmico às requisições e *collections*.
- ★ Alguns conceitos importantes:
  - ○ *Collection*: Conjunto de requisições com objetivo manter espaço organizado, colaborar com times, gerar documentações e executar testes
  - ○ *Pre-request script*: É executado antes de uma requisição ser enviada ao servidor.
  - ○ *Test script*: É executado após o envio da requisição e recebimento da resposta.



Sobre o Postman

- Poderoso *runtine* baseado em Node.js que permite adicionar comportamento dinâmico às requisições e *collections*.
- Alguns conceitos sobre variáveis:
  - Escopo de variáveis:
    - Global - > acesso em todo workspace
    - De ambiente -> Diferentes configurações de ambientes
    - De collection -> Específica para collection e independente de ambiente
    - De arquivos externos -> Definidas a partir de arquivos CSV ou JSON
    - Local - > Variáveis temporárias criadas e acessadas a partir dos scripts individuais e sobrescrevem todas as outras variáveis.

Ordem de execução



★ Atenção: Os *pre-request scripts* são úteis para configurar pré-condições para o testes:

- setar variáveis, parâmetros, cabeçalhos e dados do corpo da mensagem
- depuração a partir do uso de logs

Escrevendo scripts de teste

★ Observações:
- **pm** - objeto de biblioteca javascript que provê funcionalidades relacionadas à requisições/respostas
- É possível uso do comando ***require*** para importar outras bibliotecas
- Os testes podem utilizar a sintaxe da biblioteca CHAI, com uso do BDD.
- Os testes podem utilizar o conceito de ***expect***

Expect / Should - Chai (chaijs.com)

Assertion Styles - Chai (chaijs.com)

Test script examples | Postman Learning Center

Postman JavaScript reference | Postman Learning Center

Endpoints a serem testados

- User
  - POST - Create User
  - GET - Login
  - GET - Get user by username
  - DELETE - Delete a user
- Pet
  - POST - Add Pet to the store
  - PUT - Update PET
  - DELETE - Delete a pet

Desafios

- User
  - POST - Create User
  - GET - Login
  - GET - Get user by username
  - DELETE - Delete a user
- Pet
  - POST - Add Pet to the store
  - PUT - Update PET
  - DELETE - Delete a pet

Para fazer:
1 - Que tal finalizar os testes dos outros endpoints de user e exportar a *collection ou compartilhar com algum colega*?
2- Importar a *collection* com PETs e adicionar novas validações.

Desafios:
1 - Que tal utilizar outra ferramenta para testar esses endpoints e comparar com o Postman?
Insomnia, Hoppscotch,...

Aula 3

Testes automatizados de API com Rest Assured

// Testes de API: dos manuais aos automatizados

Objetivos

1. Criar e configurar projeto de automação com Rest Assured
2. Entender e criar estrutura do projeto com abordagem híbrida
3. Implementar scripts automatizados para endpoints de Usuário

Links importantes

DiUS/java-faker: Brings the popular ruby faker gem to Java (github.com)

JUnit 5 User Guide

The best way to add a Request Body to a POST request using Rest-Assured - DEV Community 👩‍💻👨‍💻

Botando a mão na massa!

TO DO: Para essa aula sua tarefa será concluir a automação dos *endpoints de* PETS!

- ➔ Use e abuse da documentação do REST ASSURED e seja feliz!
- ➔ Demonstre sua nova skill ou algo que aprendeu com essa nova exploração!

Testes de API: dos manuais aos automatizados

Carolina Santana Louzada

Analista QA - Venturus

Percurso

Aula 1 A API no mundo da qualidade de software

Aula 2 Testes manuais e automatizados com Postman

Aula 3 Testes automatizados com Rest Assured

Dúvidas durante o curso?

> Fórum do curso
> LinkedIn: Carolina Santana Louzada
> Comunidade online (Discord)

SCAN ME